



UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/651,376	08/29/2003	Shubhendu S. Mukherjee	42P15452	9255

8791 7590 04/03/2006

BLAKELY SOKOLOFF TAYLOR & ZAFMAN
12400 WILSHIRE BOULEVARD
SEVENTH FLOOR
LOS ANGELES, CA 90025-1030

EXAMINER

EHNE, CHARLES

ART UNIT PAPER NUMBER

2113

DATE MAILED: 04/03/2006

Please find below and/or attached an Office communication concerning this application or proceeding.

Office Action Summary	Application No. 10/651,376	Applicant(s) MUKHERJEE ET AL.	
	Examiner Charles Ehne	Art Unit 2113	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

- 1) ☒ Responsive to communication(s) filed on 29 August 2003.
- 2a) ☐ This action is **FINAL**. 2b) ☒ This action is non-final.
- 3) ☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

- 4) ☒ Claim(s) 1-29 is/are pending in the application.
- 4a) Of the above claim(s) _____ is/are withdrawn from consideration.
- 5) ☐ Claim(s) _____ is/are allowed.
- 6) ☒ Claim(s) 1-29 is/are rejected.
- 7) ☐ Claim(s) _____ is/are objected to.
- 8) ☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

- 9) ☒ The specification is objected to by the Examiner.
- 10) ☐ The drawing(s) filed on _____ is/are: a) ☐ accepted or b) ☐ objected to by the Examiner.
Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).
Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).
- 11) ☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

- 12) ☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).
- a) ☐ All b) ☐ Some * c) ☐ None of:
1. ☐ Certified copies of the priority documents have been received.
 2. ☐ Certified copies of the priority documents have been received in Application No. _____.
 3. ☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

- | | |
|---|---|
| 1) <input checked="" type="checkbox"/> Notice of References Cited (PTO-892) | 4) <input type="checkbox"/> Interview Summary (PTO-413)
Paper No(s)/Mail Date. _____ |
| 2) <input type="checkbox"/> Notice of Draftsperson's Patent Drawing Review (PTO-948) | 5) <input type="checkbox"/> Notice of Informal Patent Application (PTO-152) |
| 3) <input checked="" type="checkbox"/> Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08)
Paper No(s)/Mail Date _____ | 6) <input type="checkbox"/> Other: _____ |

DETAILED ACTION

Specification

Applicant is reminded of the proper content of an abstract of the disclosure.

A patent abstract is a concise statement of the technical disclosure of the patent and should include that which is new in the art to which the invention pertains. If the patent is of a basic nature, the entire technical disclosure may be new in the art, and the abstract should be directed to the entire disclosure. If the patent is in the nature of an improvement in an old apparatus, process, product, or composition, the abstract should include the technical disclosure of the improvement. In certain patents, particularly those for compounds and compositions, wherein the process for making and/or the use thereof are not obvious, the abstract should set forth a process for making and/or use thereof. If the new technical disclosure involves modifications or alternatives, the abstract should mention by way of example the preferred modification or alternative.

The abstract should not refer to purported merits or speculative applications of the invention and should not compare the invention with the prior art.

Where applicable, the abstract should include the following:

- (1) if a machine or apparatus, its organization and operation;
- (2) if an article, its method of making;
- (3) if a chemical compound, its identity and use;
- (4) if a mixture, its ingredients;
- (5) if a process, the steps.

Extensive mechanical and design details of apparatus should not be given.

The abstract of the disclosure is objected to because of its lack of disclosing the present invention. Correction is required. See MPEP § 608.01(b).

Claim Rejections - 35 USC § 103

The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

- (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and

Art Unit: 2113

the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negated by the manner in which the invention was made.

The factual inquiries set forth in *Graham v. John Deere Co.*, 383 U.S. 1, 148

USPQ 459 (1966), that are applied for establishing a background for determining obviousness under 35 U.S.C. 103(a) are summarized as follows:

1. Determining the scope and contents of the prior art.
2. Ascertaining the differences between the prior art and the claims at issue.
3. Resolving the level of ordinary skill in the pertinent art.
4. Considering objective evidence present in the application indicating obviousness or nonobviousness.

Claims 1-2 and 4-29 are rejected under 35 U.S.C. 103(a) as being unpatentable over Mukherjee (US 2001/0034854) taken in view of Ando (6,519,730).

As to claim 1, Mukherjee discloses a method comprising:

executing corresponding instruction threads as a leading thread and a trailing thread (page 4, ¶ 0039, lines 7-11); and

comparing the result from the selected instruction executed in the leading thread to the result from the selected instruction executed in the trailing thread (column 4, ¶ 0041, lines 8-10).

However Mukherjee discloses if the compare circuit determines that there has been a mismatch a fault recovery scheme is initiated and the processors threads are restarted at a point before the fault occurred (page 4, ¶ 0041). Mukherjee does not disclose:

saving a processor state corresponding to execution of a selected instruction in a history buffer before writing a result from the selected instruction to a destination register; and

restoring the processor state corresponding to a previous instruction using data from the history buffer if the comparison indicates a fault.

Ando discloses a method of restoring the state of a processor to an earlier state before the fault was detected (column 3, lines 53-57). Ando does disclose:

saving a processor state corresponding to execution of a selected instruction in a history buffer before writing a result from the selected instruction to a destination register (column 4, lines 8-12); and

restoring the processor state corresponding to a previous instruction using data from the history buffer if the comparison indicates a fault (column 4, lines 13-17).

It would have been obvious to one of ordinary skill in this art at the time of invention by applicant to implement Ando's method of saving the processors state before writing a result to a destination register and restoring the processor state corresponding to a previous instruction with Mukherjee's fault detection and recovery method. A person of ordinary skill in the art would have been motivated to make the modification because by saving the state of the processor the system is able to recover from an intermittent failure when an error is detected by restoring the state of the processor and re-execute a sequence of instructions (Ando: column 2, lines 35-43).

As to claim 2, Mukherjee discloses the method of claim 1 wherein the leading thread and the trailing thread are executed by a single processor (page 4, ¶ 0039, lines 7-11).

As to claim 4, Ando discloses the method of claim 1 wherein the processor state is stored in an entry in the history buffer that stores an instruction pointer to the selected

Art Unit: 2113

instruction, a value stored in the destination register, wherein the value in the destination register is to be overwritten by the result of the selected instruction, and a register map that indicates a mapping of one or more architectural registers to one or more physical registers (columns 3-4, lines 64-17).

As to claim 5, Ando discloses the method of claim 1 wherein restoring the processor state corresponding to a previous instruction if the comparison indicates a fault comprises:

- selectively flushing results of instructions that started execution after an instruction causing the fault started execution (column 4, lines 51-59);

- restoring architectural state to a checkpoint corresponding to a state at which the instruction causing the fault started execution, wherein at least a portion of the restored architectural state is retrieved from the history buffer (columns 3-4, lines 53-12).

As to claim 6, Ando discloses the method of claim 5 wherein selectively flushing results of instructions that started execution after an instruction causing the fault started execution comprises:

- flushing non-retired speculative instructions from the leading thread execution circuitry (column 6, lines 37-39 & column 4, lines 51-59);

- flushing an architectural state of the trailing thread from the trailing thread execution circuitry (column 6, lines 37-39 & column 4, lines 51-59); and

- flushing the history buffer after register values used to restore the architectural state to the checkpoint are retrieved (column 4, lines 8-17).

As to claim 7, Ando discloses the method of claim 5 wherein the checkpoint corresponds to the architectural state at a time at which execution of the instruction causing the fault is started (column 4, lines 8-12).

As to claim 8, Ando discloses the method of claim 5 wherein the checkpoint corresponds to the architectural state at a time prior to which execution of the instruction causing the fault is started (column 4, lines 8-12).

As to claim 9, Ando discloses the method of claim 5 wherein restoring architectural state to a checkpoint corresponding to a state at which the instruction causing the fault started execution, wherein at least a portion of the restored architectural state is retrieved from a history buffer comprises:

retrieving previous register values for registers written by or subsequent to the instruction causing the fault from the history buffer for the leading thread (column 4, lines 62-67);

restoring the previous register values to provide a restored architectural state for the leading thread (column 4, lines 62-67); and

copying the restored architectural state for the leading thread to an architectural register file for the trailing thread to provide a restored architectural state for the trailing thread (column 6, lines 37-44 & columns 3-4 and lines 64-4)).

As to claim 10, Mukherjee discloses the method of claim 1 wherein the selected instruction comprises a branch instruction (page 3, ¶ 0033).

As to claim 11, Mukherjee discloses an apparatus comprising:

means for executing corresponding threads as a leading thread and a trailing thread (page 4, ¶ 0039, lines 7-11); and

means for comparing the result from the selected instruction executed in the leading thread to the result from the selected instruction executed in the trailing thread (column 4, ¶ 0041, lines 8-10);

However Mukherjee discloses if the compare circuit determines that there has been a mismatch a fault recovery scheme is initiated and the processors threads are restarted at a point before the fault occurred (page 4, ¶ 0041). Mukherjee does not disclose:

means for saving a processor state corresponding to execution of a selected instruction before writing a result from the selected instruction to a destination register; and

means for restoring the processor state corresponding to a previous instruction if the comparison indicates a fault.

Ando discloses a method of restoring the state of a processor to an earlier state before the fault was detected (column 3, lines 53-57). Ando does disclose:

means for saving a processor state corresponding to execution of a selected instruction before writing a result from the selected instruction to a destination register (column 4, lines 8-12); and

means for restoring the processor state corresponding to a previous instruction if the comparison indicates a fault (column 4, lines 13-17).

It would have been obvious to one of ordinary skill in this art at the time of invention by applicant to implement Ando's method of saving the processors state before writing a result to a destination register and restoring the processor state corresponding to a previous instruction with Mukherjee's fault detection and recovery method. A person of ordinary skill in the art would have been motivated to make the modification because by saving the state of the processor the system is able to recover from an intermittent failure when an error is detected by restoring the state of the processor and re-execute a sequence of instructions (Ando: column 2, lines 35-43).

As to claim 12, Ando discloses the apparatus of claim 11 wherein the means for restoring the processor state corresponding to a previous instruction if the comparison indicates a fault comprises:

means for selectively flushing results of instructions that started execution after an instruction causing the fault started execution (column 4, lines 51-59);

means for restoring architectural state to a checkpoint corresponding to a state at which the instruction causing the fault started execution, wherein at least a portion of the restored architectural state is retrieved from a history buffer (columns 3-4, lines 53-12).

As to claim 13, Ando discloses the method of claim 11 wherein the means for selectively flushing results of instructions that started execution after an instruction causing the fault started execution comprises:

means for flushing non-retired speculative instructions from a thread having the instruction that caused the fault (column 6, lines 37-39 & column 4, lines 51-59);

Art Unit: 2113

means for flushing an architectural state of a trailing thread having the instruction that caused the fault (column 6, lines 37-39 & column 4, lines 51-59); and

means for flushing a history buffer after register values used to restore the architectural state to the checkpoint are retrieved (column 4, lines 8-17).

As to claim 14, Mukherjee discloses an apparatus comprising:

leading thread execution circuitry to execute a leading thread of instructions (page 4, ¶ 0039, lines 7-11); and

trailing thread execution circuitry to execute a trailing thread of instructions (page 4, ¶ 0039, lines 7-11).

However Mukherjee discloses if the compare circuit determines that there has been a mismatch a fault recovery scheme is initiated and the processors threads are restarted at a point before the fault occurred (page 4, ¶ 0041). Mukherjee does not disclose:

a history buffer coupled with the leading thread execution circuitry and the trailing thread execution circuitry to store information related to execution of a selected instruction from the leading thread of instructions, wherein the information stored in the history buffer is used to restore an architectural state corresponding to a checkpoint if an execution fault is detected.

Ando discloses a method of restoring the state of a processor to an earlier state before the fault was detected (column 3, lines 53-57). Ando does disclose:

a history buffer coupled with the leading thread execution circuitry and the trailing thread execution circuitry to store information related to execution of a selected

Art Unit: 2113

instruction from the leading thread of instructions, wherein the information stored in the history buffer is used to restore an architectural state corresponding to a checkpoint if an execution fault is detected (column 4, lines 5-18).

It would have been obvious to one of ordinary skill in this art at the time of invention by applicant to implement Ando's history buffer with Mukherjee's fault detection and recovery method. A person of ordinary skill in the art would have been motivated to make the modification because the history buffer provides the last valid checkpoint and the current state, which allows for restoration of a previous valid state to re-execute instructions (column 4, lines 5-18).

As to claim 15, Ando discloses the apparatus of claim 14 wherein the history buffer stores an instruction pointer to the selected instruction, a value stored in the destination register, wherein the value in the destination register is to be overwritten by the result of the selected instruction, and a register map that indicates a mapping of architectural registers to physical registers (column 4, lines 5-12).

As to claim 16, Ando discloses the apparatus of claim 15 wherein the architectural state corresponding to the checkpoint is restored by selectively flushing results of instructions that started execution after an instruction causing the fault started execution and restoring architectural state to a checkpoint corresponding to a state at which the instruction causing the fault started execution, wherein at least a portion of the restored architectural state is retrieved from the history buffer (column 6, lines 37-39 & column 4, lines 51-59).

As to claim 17, Ando discloses the apparatus of claim 16 wherein the architectural state corresponding to the checkpoint is restored by flushing non-retired speculative instructions from the execution circuitry corresponding to the thread having an instruction that caused the fault, flushing an architectural state of the execution circuitry corresponding to the thread having the instruction that caused the fault, and flushing the history buffer after register values used to restore the architectural state to the checkpoint are retrieved (column 6, lines 37-39 & column 4, lines 51-59).

As to claim 18, Mukherjee discloses the apparatus of claim 14 wherein the execution fault is caused by a branch instruction (page 3, ¶ 0033).

As to claim 19, Ando discloses the apparatus of claim 14 wherein the checkpoint corresponds to the architectural state at a time at which an instruction causing the fault is started (column 4, lines 8-12).

As to claim 20, Ando discloses the apparatus of claim 14 wherein the checkpoint corresponds to the architectural state at a time prior to which an instruction causing the fault is started (column 4, lines 8-12).

As to claim 21, Mukherjee discloses the apparatus of claim 14 wherein the leading thread execution circuitry and the trailing thread execution circuitry are part of a single processor (page 4, ¶ 0039, lines 7-11).

As to claim 22, Ando discloses the apparatus of claim 14 wherein the leading thread execution circuitry is part of a first processor and the trailing thread execution circuitry are part of a second processor (column 6, lines 13-19). Assuming that the first

Art Unit: 2113

computer is processing the Mukherjee's leading thread and the second computer is processing the trailing thread.

As to claim 21, Mukherjee discloses a system comprising:

leading thread execution circuitry to execute a leading thread of instructions (page 4, ¶ 0039, lines 7-11);

trailing thread execution circuitry to execute a trailing thread of instructions (page 4, ¶ 0039, lines 7-11); and

an input/output controller coupled with the leading thread execution circuitry (Figure 1.93, page 3, ¶ 0031, line 7).

However Mukherjee discloses if the compare circuit determines that there has been a mismatch a fault recovery scheme is initiated and the processors threads are restarted at a point before the fault occurred (page 4, ¶ 0041). Mukherjee does not disclose:

a history buffer coupled with the leading thread execution circuitry and the trailing thread execution circuitry to store information related to execution of a selected instruction from the leading thread of instructions, wherein the information stored in the history buffer is used to restore an architectural state corresponding to a checkpoint if an execution fault is detected.

Ando discloses a method of restoring the state of a processor to an earlier state before the fault was detected (column 3, lines 53-57). Ando does disclose:

a history buffer coupled with the leading thread execution circuitry and the trailing thread execution circuitry to store information related to execution of a selected

Art Unit: 2113

instruction from the leading thread of instructions, wherein the information stored in the history buffer is used to restore an architectural state corresponding to a checkpoint if an execution fault is detected (column 4, lines 5-18).

It would have been obvious to one of ordinary skill in this art at the time of invention by applicant to implement Ando's history buffer with Mukherjee's fault detection and recovery method. A person of ordinary skill in the art would have been motivated to make the modification because the history buffer provides the last valid checkpoint and the current state, which allows for restoration of a previous valid state to re-execute instructions (column 4, lines 5-18).

As to claim 24, Ando discloses the system of claim 23 wherein the history buffer stores an instruction pointer to the selected instruction, a value stored in the destination register, wherein the value in the destination register is to be overwritten by the result of the selected instruction, and a register map that indicates a mapping of architectural registers to physical registers (column 4, lines 5-12).

As to claim 25, Ando discloses the system of claim 24 wherein the architectural state corresponding to the checkpoint is restored by selectively flushing results of instructions that started execution after an instruction causing the fault started execution and restoring architectural state to a checkpoint corresponding to a state at which the instruction causing the fault started execution, wherein at least a portion of the restored architectural state is retrieved from the history buffer (column 6, lines 37-39 & column 4, lines 51-59).

As to claim 26, Ando discloses the system of claim 25 wherein the architectural state corresponding to the checkpoint is restored by flushing non-retired speculative instructions from the execution circuitry corresponding to the thread having an instruction that caused the fault, flushing an architectural state of the execution circuitry corresponding to the thread having the instruction that caused the fault, and flushing the history buffer after register values used to restore the architectural state to the checkpoint are retrieved (column 6, lines 37-39 & column 4, lines 51-59).

As to claim 27, Mukherjee discloses the system of claim 23 wherein the execution fault is caused by a branch instruction (page 3, ¶ 0033).

As to claim 28, Ando discloses the system of claim 23 wherein the checkpoint corresponds to the architectural state at a time at which an instruction causing the fault is started (column 4, lines 8-12).

As to claim 29, Ando discloses the system of claim 23 wherein the leading thread execution circuitry is part of a first processor and the trailing thread execution circuitry are part of a second processor (column 6, lines 13-19). Assuming that the first computer is processing the Mukherjee's leading thread and the second computer is processing the trailing thread.

Claim 3 is rejected under 35 U.S.C. 103(a) as being unpatentable over Mukherjee and Ando as applied to claim 1 above, and further in view of Applicants admitted prior art: Detailed design and Evaluation of Redundant Multithreading Alternatives (AAPA).

Art Unit: 2113

As to claim 3, Mukherjee and Ando disclose executing corresponding instructions threads as leading thread and trailing thread on one processor (see claim rejection 1). The combination fails to disclose wherein the leading thread and the trailing thread are executed by multiple processors.

AAPA discloses a chip level redundant threading (CRT) technique. AAPA does disclose the method of claim 1 wherein the leading thread and the trailing thread are executed by multiple processors (Figure 5, page 2, lines 19-33).

It would have been obvious to one of ordinary skill in this art at the time of invention by applicant to implement having wherein the leading thread and the trailing thread are executed by multiple processors. A person of ordinary skill in the art would have been motivated to make the modification because by running the leading and trailing threads on multiple processors you achieve lockstepping's permanent fault coverage while maintaining the low overhead output comparison and efficiency of a single redundant threaded processor (AAPA: page 7, lines 20-27).

Conclusion


Any inquiry concerning this communication or earlier communications from the examiner should be directed to Charles Ehne whose telephone number is (571)-272-2471. The examiner can normally be reached on Monday-Friday 8:30-5:00.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Robert Beausoliel can be reached on (571)-272-3645. The fax phone

Art Unit: 2113

number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see <http://pair-direct.uspto.gov>. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).


ROBERT BEAUSOLIEL
SUPERVISORY PATENT EXAMINER
TECHNOLOGY CENTER 2100